

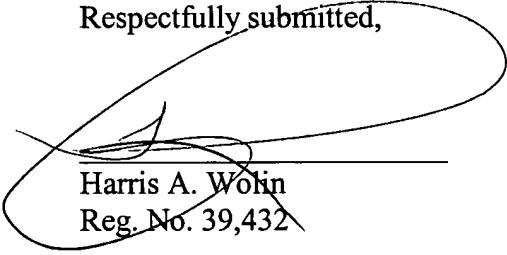
Patent 5,963,938). Finally, claim 10 is rejected under 35 U.S.C. §103(a) as being unpatentable over Wasilewski '275 in view of Bowman et al. '225 and further in view of Carey et al. (U.S. Patent 5,905,982).

Applicant Andreas Rippich conducted a telephone interview with the Examiner on September 10, 2003. As a result of such interview, it was agreed that the Examiner would consider a substantive response prepared by the Applicant. Accordingly, please find enclosed (1) a letter to the Examiner dated September 17, 2003, from Applicant Andreas Rippich, along with (2) a Response to the first Office Action of 06/06/03 concerning the non-final rejection of patent application No. 09/768,727 dated September 17, 2003. It is respectfully requested that the Examiner reconsider the stated rejections in view of the arguments presented therein.

In view of the above amendments and remarks, it is believed that claims 1-25, consisting of independent claims 1, 18 and 22 and the claims dependent therefrom, are in condition for allowance. Passage of this case to allowance is earnestly solicited. However, if for any reason the Examiner should consider this application not to be in condition for allowance, the Examiner is respectfully requested to telephone the Applicant at 011-49-89-8947439 or the undersigned attorney at the number listed below prior to issuing a further Action.

Any fee due with this paper may be charged on Deposit Account 50-1290.

Respectfully submitted,


Harris A. Wolin
Reg. No. 39,432

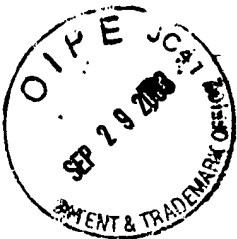
CUSTOMER NUMBER 026304

PHONE: (212) 940-8708

FAX: (212) 894-5708 (direct)

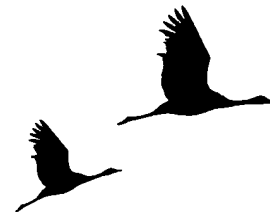
DOCKET NO.: 3134/WEICK (083615-87523)

Filed by Express Mail
(Receipt No. EL9801841mws)
on 9-29-03
pursuant to 37 C.F.R. 1.10.
by Frances Doyle



ANDRIP

Andreas Rippich
Flurstr. 2, 82110 Germering, Germany
Tel: 0049-89-8947439
Fax: 0049-89-8414331
www.andrip.de, andrip@aol.com



Andreas Rippich • Flurstr. 2 • 82110 Germering • Germany

September 17, 2003

United States Patent and Trademark Office
c.o. Examiner Mr. Joon H. Hwang
P.O. Box 1450
Alexandria, Virginia 22313-1450
U.S.A.

re: Application No. 09/768,727
Filing date 01/24/2001

RECEIVED
OCT 06 2003
Technology Center 2100

Dear Mr. Hwang,

in the course of our brief telephone conversation on September 10th we agreed that I, the inventor of the algorithm filed for patent as noted above, should answer your »Detailed Actions« of June 6th by mail via my patent attorney KMZ Rosenman.

As I mentioned to you, I see the five cited patents as

- a) providing solutions to requirements in other areas than my algorithm serves, and
- b) doing so in ways which are not congruent with my algorithm
(i.e. different requirements have led to different solutions).

In my view, therefore, § 35 U.S.C. 103(a) does not apply and I have done my best to illustrate this in the enclosed document.

I would like to ask you to review your conclusions in the light of my response. If you see the need for any further clarification, please call me (preferably during your morning hours) on 0049-89-8947439, and I will be glad to call you back. And if it is of any help I can also send you a software program demonstrating the algorithm.

With best regards,

Enclosure: Response to non-final rejection

ANDRIP

Andreas Rippich
Flurstr. 2, 82110 Germering, Germany
Tel: 0049-89-8947439
Fax: 0049-89-8414331
www.andrip.de, andrip@aol.com



Response to the first Office Action of 06/06/03 concerning the non-final rejection of patent application No. 09/768,727

author: Andreas Rippich
date: September 17, 2003

Description of the Rippich Method	2
Description of the problem and its solution	2
Example of a travel booking using an AND operator	3
Salient points of the Rippich Method as contained in the patent application	5
Response to the non-final rejection	6
Response to the rejections concerning Wasilewski	6
Response to the rejections concerning Bowman	7
Response to the rejections concerning Wilson	9
Response to the rejections concerning Carey	10
Response to the rejections concerning Reed	11

Description of the Rippich Method

The Rippich Method is applicable for searching through various types of data pools. In contrast to for example most internet search engines the user is not required to enter data on a »hit or miss« basis, but is offered at each step all possible valid choices. Since only terms fulfilling the current query are offered for selection a null value cannot occur. So empty results are by nature of the algorithm not possible.

Description of the problem and its solution

Conventional search engines work as follows: the user has to type in the words that are to be searched for and has to join them with operators like AND or NOT.

A treasure diver for example might want to go on holiday to a place where a ship loaded with valuable goods sank, hoping to find it. In order to determine his travel destination he might search the internet for a document containing the phrases »ship«, »sink« and »treasure«. After having typed in these words the search engine would start working and would after some time end up with the result »0 hits« or »no documents found«, and the user would not know why:

- One reason might be that the words are not listed in their basic form; so if the user would have typed in the perfect tense »sunk« instead of the present tense »sink« he would have got several hits, but unfortunately he does not know this.

Remark: to cope with this problem the user may not use the comparatively simple phrase "»ship« AND »sink« AND »treasure«"; instead he would have to type in the quite complex phrase "(»ship« OR »ships«) AND (»sink« OR »sank« OR »sunk«) AND (»treasure« OR »treasures«)". He could try using wildcards but when using »ship*« he would also find unwanted words like »shipowner«, »shiproute«, »shipyard« et cetera.

- Another reason might be that the words »ship«, »sink« and »treasure« occur in no document simultaneously regardless of their declension or conjugation. So the user might try the word »boat« instead of »ship«, and even this might not lead to success. The solution might be the combination »vessel«, »sink« and »treasure«, but the user would probably never find out.

So the standard problem with all present search engines is that when a query results in a null value the user has no idea why and he does not know how to alter his query in order to be successful. As a consequence the user will probably try several combinations until he is frustrated and gives up his search.

In contrast to this the Rippich Method is the users partner as he or she searches a data pool. It assumes that the data pool has been indexed and uses these indices at each interaction. Normally the indexed terms will be words (i.e. within documents) but may also be of other type (e.g. pictures). The search progresses as a sequence of interactive steps, alternating between choice of content term and choice of logical operator. As a result of each selection made by the user, the Method displays all remaining choices congruent with the current accumulation of the query. This informs the user of all his now possible options and at the same time provides the next level of choice. Thus a null response is by the nature of the algorithm not possible.

To put it another way, the Rippich Method avoids null values because

- either all at a given time valid terms (and only these)
- or all at a given time valid operators (and only these)

are alternately displayed, whilst the query is refined by selection from the displayed elements.

Example of a travel booking using an AND operator

The treasure diver mentioned above lives in New York and for the winter he wants to book a journey to the warm South at his local travel agency by internet. He first wants to visit Disney World and then he wants to go to Miami Beach where he can dive and relax.

Since the customer wants to go to Disney World he chooses Orlando (light grey background) as his first destination from the list named »available destinations«. As a consequence the three packages trip #1, trip #4 and trip #5 are offered indicating that they and only they contain the destination »Orlando«.

available destinations	available operators	list of destinations	=>	available offers
Fort Lauderdale		Orlando		trip #1
Miami				trip #4
Orlando				trip #5
Sarasota				

So far no decision has been made. So the user may scroll through the list of the available destinations and find out by marking Sarasota, that this city is contained in the two travel packages trip #5 and trip #12 and only in these.

Fort Lauderdale
Miami
Orlando
Sarasota

A realistic implementation would suppress those lists which at a given time would bear no information. So instead of ordering the lists horizontally in rows, they might be separated by bars in a vertical orientation thus fitting in the tiny display of an organizer or even of a UMTS- or i-mode cell phone.

In this first step only the available destinations are of importance. Even the available offers are of interest to the user only after he has decided on a certain destination. Thus the relevant information might be displayed on a cell phone or on an organizer as shown on the left.

After the user decides on »Orlando« by clicking (resulting in a dark grey background) the operators available for this item are then shown. In the beginning the focus (light grey background) is on a random term like »AND«:

available destinations	available operators	list of destinations	=>	available offers
Fort Lauderdale	AND	Orlando		trip #1
Orlando	OR	Orlando		trip #4
Sarasota	EXCL. OR			trip #5
	NOT			

Fort Lauderdale
Orlando
Sarasota
AND
OR
EXCL. OR
NOT

On a cell phone or organizer display the operators suitable for the chosen item »Orlando« might be listed below the available destinations, separated by a dark line.

The result of this query is immediately visible:

- The user has not taken any action since marking the operator »AND«, so in the list of available destinations the focus is still on Orlando. The resulting Boolean operation »Orlando AND Orlando« brings no change in the list of available offers still listing the three packages trip #1, trip #4 and trip #5.
- But the city »Miami« vanishes from the list of available destinations. This indicates that the wanted trip to Miami via Orlando is not on offer.

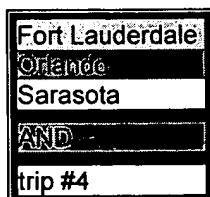
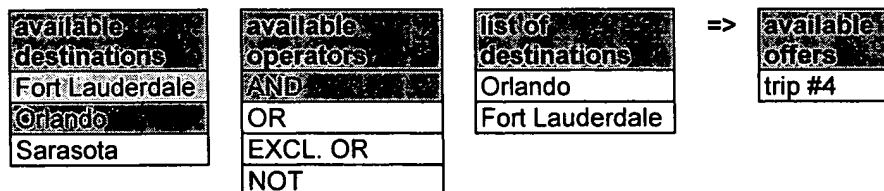
At this point the advantage of the method becomes obvious:

- Using conventional search engines the user would have to clumsily type the phrases »Orlando AND Miami« only to find out that no travel package containing these two destinations exists. He would probably give up frustrated since he would not know how to change his query to achieve a hit. Thus the travel agency would lose a customer.
- However in our example using the Rippich Method the customer does not need to type anything. Just by clicking on the terms »Orlando« and »AND« he will immediately know that the planned travel route is not offered because »Miami« vanished from the available items in response to clicking the operator »AND«.
- At the same time all other destinations that can be combined with Orlando are displayed. The user sees at a glance that the travel packages »Orlando AND Fort Lauderdale« and »Orlando AND Sarasota« and only they are being offered. Instead of giving up disappointed the customer might consider that Fort Lauderdale is just as good as Miami for him if not even better. Because on one hand Fort Lauderdale is close to Miami so he can easily visit it, and on the other hand it is known for its famous fishing grounds. So he will most probably book the trip »Orlando AND Fort Lauderdale« without regretting the change of destination.

So far no decision about the operator has been made. So scrolling through the list the user would find out that the city Orlando would vanish when marking the operator »NOT« since there cannot be any offer which does contain Orlando and at the same time does not contain Orlando. Miami however would show up again because there is no trip including the two cities Orlando and Miami.

Basically the query has successfully ended after clicking the two terms »Orlando« and »AND«. Without even having typed a single word the customer knows that the trip he had envisaged is not available, and at the same time he found an alternative he is happy with.

After deciding for the operator »AND« by clicking (resulting in a dark grey background) the focus moves to the cities available for this operator. Orlando has already been chosen, so it is displayed with a dark grey background. By clicking the chosen city »Fort Lauderdale« the query is over and the travel package »trip #4« can be booked.



In the same way the terms »Orlando« and »AND« show up with a dark grey background on a cell phone or organizer display because they had already been chosen, and »Fort Lauderdale« is displayed with a light grey background indicating that it is being focused but not yet chosen. So the customer could still point to Sarasota which might result in the travel package trip #5.

But the customer wants to go to Fort Lauderdale, so he ends the query by clicking »Fort Lauderdale«, thus choosing »trip #4« and continuing to the booking process.

Salient points of the Rippich Method as contained in the patent application

The Rippich patent application describes as an essential feature that **directly after choosing an item** (i.e. term or operator) **the displayed set is audited** (recomputed and redisplayed), eliminating all items which would produce unwanted results.

This is stated in claim 1:

»entering a search term« (for initialization, claim 1.a),
then *»entering at least one search command«* (i.e. operator and item, claim 1.b),
immediately being followed by *»auditing ... with reference to the contents of the database, and eliminating ... all impossible items or operators«* (claim 1.c),
and *»displaying ... the remaining contents«* (claim 1.d),
»wherein steps b, c, and d are repeated until ...« (end of claim 1).

This procedure is graphically illustrated in the flow chart referred to as fig. 6.

As described in claim 1.c, the Rippich Method avoids null values the following way:

- a) after choosing a term all operators which would result in a null value are no longer available, and
- b) after choosing an operator all terms which would result in a null value are no longer available.

This means that directly after each step no invalid terms or operators are displayed and available for further selection.

None of the 5 cited patents discloses this updating of the display of data immediately after any term or operator is chosen so 35 U.S.C. 103(a), cited as the basis for all rejections, does not apply.

Response to the non-final rejection

Response to the rejections concerning Wasilewski

One of the main goals of Wasilewski's invention is to enable the user to browse through the internet without the use of a keyboard.

The »Detailed Action« says that *»Wasilewski discloses auditing at least one of a displayed set in response to the chosen operator (figs. 8-11 and lines 6-59 in col. 7)«*. It is true that Wasilewski displays terms but he is silent on auditing these terms in response to the chosen operator.

So contrary to the Rippich application Wasilewski discloses that all desired terms and operators are selected **before** the search is submitted. **No auditing is performed after each selection, and thus null values may result:**

-) *»Through operation of the selection device, multiple terms from the same list may be chosen. For example, from list 702, both "cars" and "computing" may be chosen«* (lines 11-14 in col. 7)

Contrary to Wasilewski the Rippich algorithm would audit and update the list of available terms immediately after »cars« had been chosen, so the term »computing« would be deleted from the list of terms if the query »cars AND computing« would lead to a null value.

-) *»As more terms are associated with a single term, its representation is contemplated to change from a single representation to a representation followed by periods of ellipsis«* (lines 16-19 in col. 7)

There is no auditing and no updating even in complex queries.

-) *»At this point a user submits the search«* (lines 36-37 in col. 7)

Only after having entered the whole query the user submits the search. Since there was no updating in between a null value may result.

-) *»In this example, the terms contained in pick list 901 include "Acura", "Audi", "BMW", "Buick", "Cadillac", and "Toyota". From the new pick list 901, a user may select any of these terms (and, by following associated terms along, select additional terms as well).«* (lines 55-59 in col. 7)

In this example there also is no updating before selecting additional terms.

Wasilewski is silent on auditing and updating the lists of available terms/operators immediately after having selected an operator/term whilst this is vital to the Rippich Method, so this prior art does not go in the direction of the Rippich application.

Response to the rejections concerning Bowman

The »Detailed Action« says that »Bowman discloses eliminating impossible items, which would produce an unwanted result at the end of a database search, and auditing and eliminating being performed by an automatic auditing processor (line 28 in col. 2 thru line 30 in col. 3, lines 32-35 in col. 4, lines 42-55 in col. 9, and lines 42-67 in col. 12) in order to prevent a NULL query result«.

It is true that he offers an algorithm to prevent NULL query results, but unlike Rippich he does it not by »eliminating impossible items, which would produce an unwanted result at the end of a database search, and auditing and eliminating being performed by an automatic auditing processor«; instead, he uses query log files based on former successful queries.

Accordingly, there is no auditing in the sense of feature c of claim 1 of the Rippich application which calls for an **elimination** of items and operators which would produce an unwanted result (i.e. null value), but a presentation of **additional** search terms in combination with the search term entered by the user, which in conclusion would lead to a successful search. These additional search terms are derived from historical search data, i.e. the query log file mentioned in the Bowman document:

-) »... the related terms are generat[ed] using query term correlation data that is based on historical query submissions to the search engine.« (lines 31-33 in col. 3);
-) »Each related terms list contains the terms which have historically appeared together (in the same query) with the respective key term with the highest degree of frequency, ignoring unsuccessful query submissions (query submissions which produced a NULL query result).« (line 66 in col. 2 thru line 3 in col. 3);
-) »As indicated above, the generation process 136 parses the daily query log file in step 410 to identify and extract successful multi-term queries« (lines 42-44 in col. 9).

The fact that only former successful queries are included in the correlation data restricts the user to selecting those terms which have accumulated over the past, so **contrary to the Rippich application the user is offered only a subset of all possible successful terms**.

Bowman avoids null values by using the said historical search data instead of using Rippichs algorithm of eliminating unfitting items and operators:

- 1) Bowman states, that »the scope of the present invention is defined only by reference to the appended claims« (lines 49-51 in column 14).
- 2) Claim 1 reads:
 - (a) »processing search queries submitted to the search engine by a plurality of users over a period of time to generate query term correlation data, the query term correlation data reflecting frequencies with which query terms appear together within the same search query«;
The Rippich Method does not use any »query term correlation data« having been submitted by many users and reflecting frequencies with which query terms appear together within the same search query.
 - (c) »using at least the query term correlation data to identify a plurality of additional query terms that are deemed to be related to the at least one query term«;
The Rippich Method does not use »query term correlation data« to identify a plurality of additional query terms
 - (d) »presenting the plurality of additional query terms to the user for selection to allow the user to refine the search query«;
Bowman presents »the plurality of additional query terms to the user« and thus

expands the search by offering those terms which have not led to null values in the past, whilst Rippich narrows the search by eliminating all impossible terms directly after entering any operator.

- 3) All claims describing how to avoid null values are based on claim 1:
-) claim 4 is based on claim 3 which is based on claim 1
 -) claim 7 is based on claim 6 which is based on claim 1
 -) claim 19 is based on claim 14 which is similar to claim 1 since it describes a system for generating related terms also based on historical query submissions
 -) claim 21 is based on claim 20 being based on claim 19, and claim 19 is based on claim 14 which is similar to claim 1 since it describes a system for generating related terms also based on historical query submissions
 -) claim 24 is based on claim 23 being based on claim 22 which is similar to claim 1 since it also uses historical search query data
 -) claim 25 only increases »a likelihood that a modified query will not produce a NULL query result«, and it is similar to claim 1 in also using historical query information.

There are several restrictions arising from the use of historical data. Three prominent ones are:

-) The avoidance of null values is guaranteed only if the historical data are still valid and still reflect the contents of the data base:

»The related terms which remain are terms which have previously appeared, in at least one successful query submission, in combination with every term of the present query. Thus, assuming items have not been deleted from the database being searched, any of these related terms can be individually added to the present query while guaranteeing that the modified query will not produce a NULL query result.« (lines 13-20 in col. 3).

The Rippich algorithm allows the index of the database to be updated any time guaranteeing the avoidance of null values, whilst the Bowman algorithm results in the need to keep the database unchanged for a period of one or even several weeks, because its query correlation table is built up from a number of daily log files (see Fig. 6 illustrating a query correlation table based on data from Feb 1 to Feb 8).

-) The Bowman algorithm, as described above, **offers the user only a subset of the variety of possible successful terms**, whilst the Rippich algorithm on the other hand displays all possible terms.
-) Bowman avoids null values in multiple-term queries by taking the intersection between different lists, or, in other words, by deleting the terms that are not common to all lists:

»To generate a set of related terms for refining a submitted query (the "present query"), the related terms list for each term in the present query is initially obtained from the correlation data structure. If this step produces multiple related terms lists (as in the case of a multiple-term query), the related terms lists are preferably combined by taking the intersection between these lists (i.e., deleting the terms that are not common to all lists).« (lines 6-13 in col. 3).

This limits the variety of operators and generally forbids for example the use of the NOT-operator, whilst the Rippich algorithm offers the NOT-operator as long as it does not lead to a null value.

The Bowman algorithm for avoiding null values is based on query log files arising from former successful queries, whilst the Rippich algorithm for avoiding null values is based on updating and elimination of items and operators, so this prior art does not go in the direction of the Rippich application.

Response to the rejections concerning Wilson

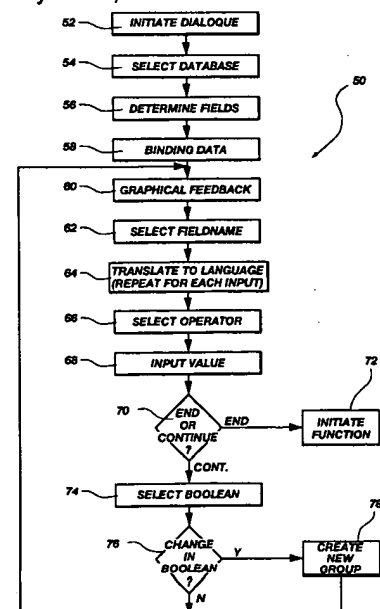
It is true that »Wilson discloses limiting possible (logical) operators for a particular argument«, but he does it by referring to field 310 by saying »... it may be advantageous to limit the possible logical operators ...« (lines 9-10 in col. 10), followed by »It may be helpful to think of the fields 310 more in terms of field name or field type since text may be handled differently than dates, which may be handled differently than accounting numbers, for example.« (lines 11-15 in col. 10), so the aim of limiting the possible logical operators is obviously to prevent illogical operations like applying the operator »divided by« to a text string or applying the operator »Change to capital letters« to a numerical value.

Wilson discloses an automatic context-organizing query interface, but he does not even aim at suppressing all invalid terms and operators after each step during the query.

This becomes obvious by the following facts:

- a) he only suggests to limit the possible logical operators, but he does not make it mandatory;
- b) he suggests to limit the possible logical operators but he nowhere limits the possible items, so he cannot prevent null values if the operations become complex;
- c) the reason he suggests to limit the possible logical operators is to prevent unlogical operations within the given filter like
 -) applying the operator »divided by« to a text string or
 -) applying the operator »Change to capital letters« to a numerical value but not to prevent null values in complex operations;
- d) he nowhere even mentions the term »null value«;
- e) **just like Wasilewski Wilson discloses that all desired terms and operators are selected before the search is submitted. No auditing is being performed after each selection, so null values may result.** This becomes even more obvious by the fact that all of Wilsons figures of filters have »OK«- and »Reset«-buttons indicating that the search is started after all terms and operators are entered; since this may result in null values in complex filters the Rippich Method demands that an auditing is performed immediately after selecting any operator or any item;
- f) the above said becomes even clearer in the flow chart as seen in fig. 2, where Wilson initiates the function (72) only after having selected all desired inputs (62, 66, 68, 74) instead of updating the values after each input:

Wilson discloses that all desired terms and operators are selected before the search is submitted, so this prior art does not go in the direction of the Rippich application, because no auditing is being performed after each selection.



Response to the rejections concerning Carey

It is true that »Carey discloses a SQL query for searching a database including relational and object oriented«, but he achieves his goal by a totally different way:

- a) The Carey patent is limited to object oriented data: »A method, apparatus, and article of manufacture for handling NULL values in SQL queries over object oriented data.« (first sentence of abstract and lines 2-3 in column 6);
- b) Also the Carey method is restricted to queries that involve subqueries: »A two-phase method is used to enable a query evaluator in a two-valued logic environment to properly handle occurrences of NULL values for predicates that involve subqueries, ...« (second sentence of abstract and lines 3-6 in column 6);
- c) To avoid NOTs he inverts basic comparators whilst Rippich does not make them available if they would lead to a NULL value: »For basic subquery predicates, negation reduction is performed by applying logical equivalence rules and inverting basic comparators (e.g., transforming < to >=) to eliminate NOTs.« (lines 8-11 in column 6);
- d) »Then, transformations are employed for the resulting positive predicates to include NULL value testing, i.e., NULL protection.«, whilst Rippich does not transform any result because there is no need for it (lines 11-13 in column 6);
- e) A similar procedure is necessary for quantified subquery predicates: »For quantified subquery predicates, in addition to performing negation reduction, quantified subqueries are converted into existential subqueries.« (lines 13-15 in column 6);

The need for such negation, transformation and conversion arises from the fact that Carey interprets the SQL query after the user has entered it; all of these problems are avoided in the Rippich application by analysing the SQL query whilst it is entered and eliminating all terms and operators that would lead to a NULL value.

In other words, with the Rippich application the user does not enter a (complex) SQL query, rather he creates it in a series of interactions with the system, which at each step only offers him possible choices based on the query so far.

Carey interprets an SQL query after the user has entered it, whilst the Rippich algorithm analyses it whilst it is created, so this prior art does not go in the direction of the Rippich application.

Response to the rejections concerning Reed

Just like Bowman Reed relies on a history of successful queries:

»Accordingly, it is an advantage of the present invention to provide a computerized system that interactively enables a user to build a technical thesaurus using on-line databases. It is another advantage of the present invention to provide a computerized system that iteratively prompts a user to interactively modify an "initial query" into a "crafted query" while searching at least one on-line database.« (first sentence of the »Summary of the invention«, lines 28-32 in page 1);

He uses these *»to build a technical thesaurus using on-line databases«* and does this by interactively modifying *»an "initial query" into a "crafted query" while searching at least one on-line database«.*

Reed is silent on auditing the database during the query and thus he is silent on avoiding null values, so this prior art does not go in the direction of the Rippich application.